Temporally Coherent Clustering of Student Data

Severin Klingler
Department of Computer Science
ETH Zurich, Switzerland
kseverin@inf.ethz.ch

Tanja Käser
Department of Computer Science
ETH Zurich, Switzerland
kaesert@inf.ethz.ch

Barbara Solenthaler
Department of Computer Science
ETH Zurich, Switzerland
sobarbar@inf.ethz.ch

Markus Gross
Department of Computer Science
ETH Zurich, Switzerland
grossm@inf.ethz.ch

ABSTRACT

The extraction of student behavior is an important task in educational data mining. A common approach to detect similar behavior patterns is to cluster sequential data. Standard approaches identify clusters at each time step separately and typically show low performance for data that inherently suffer from noise, resulting in temporally inconsistent clusters. We propose an evolutionary clustering pipeline that can be applied to learning data, aiming at improving cluster stability over multiple training sessions in the presence of noise. Our model selection is designed such that relevant cluster evolution effects can be captured. The pipeline can be used as a black box for any intelligent tutoring system (ITS). We show that our method outperforms previous work regarding clustering performance and stability on synthetic data. Using log data from two ITS, we demonstrate that the proposed pipeline is able to detect interesting student behavior and properties of learning environments.

Keywords

Evolutionary Clustering, Markov Chains, Sequence Mining, Distance Metrics

1. INTRODUCTION

The extraction of student properties is a central element in educational data mining. On the one hand, the identification of student abilities and behavior patterns allows us to draw conclusions about human learning. On the other hand, the extracted properties can be used to improve the adaptation of the underlying intelligent tutoring system (ITS).

Clustering of sequential data is a common approach to detect similar behavior patterns and has been successfully applied to a variety of applications such as reading comprehension [22], online collaboration tools [24], table-top environments [19], web browsing [25], physics simulations [4] or homework assignments [11]. Furthermore, a variety of different student behavior has been investigated. [20] identified students that impose challenges for the student models. Other work studied the relation between interaction patterns and the performance of students [3, 14] and the relation between student action sequences and their affective states [3].

Common techniques for the analysis of sequential data include sequence mining [1, 19], differential pattern mining [11] or Hidden Markov models (HMM) [5, 6]. Sequential pattern mining techniques have been contextualized using piecewise linear segmentation [14]. Others have employed semi-supervised graph clustering using the predictions from a student model as additional constraints [20]. Clustering sequential data employing similarity measures on state sequences was used in [4, 8]. These state sequences can be aggregated into Markov Chains modeling the state transitions [17]. HMM have been employed to extract stable groups from temporal data by joint optimization of the model parameters and the cluster count [18].

While the previous work discussed above analyze student clusters at a given point in time, a temporal analysis would allow to identify how interaction patterns change over time and how groups of similar students evolve. Temporal effects of cluster evolution have been analyzed in [15], based on static clustering at each time step. Static approaches are sensitive to noise in the data and may result in temporally inconsistent clusters. Evolutionary clustering methods [7] address this problem as they consider multiple subsequent time steps. The temporal smoothing increases the resulting cluster stability notably and allows for a better analysis of the clusters, i.e., the student properties and interaction patterns. Recently, an evolutionary clustering approach called AFFECT [27] has been introduced that smooths proximities of students over time followed by static clustering. AFFECT was shown to outperform static clustering algorithms.

In this paper, we present a complete processing pipeline for evolutionary clustering that can be used as a black box for any ITS. We incorporate a variation of the AFFECT method into our pipeline and demonstrate that temporal smoothing has beneficial properties for extracting student behavior and groups from educational data. We propose several extensions of the original method tailored towards learning data. Our approach is articulated in four steps. In a first step, we extract action sequences from ITS log data and aggregate them using Markov Chains. We show that the Markov Chain representation of the actions is superior to direct sequence mining techniques [4, 17] with respect to noise cancellation and the ability to identify groups of students with similar behavior. The second step consists of computing pairwise similarities between the Markov Chains. While the proposed pipeline provides flexibility in the choice of similarity measure, the Hellinger distance outperforms other metrics that

are frequently used in the educational data mining literature [4, 17]. Based on the obtained similarities, evolutionary clustering [27] is performed in the third step. The temporal aspect of the student data leads to changing behavior patterns, i.e., we expect the number of clusters and cluster sizes to change over time. Therefore, capturing cluster evolution events, such as merging, splitting, dissolving and forming of clusters, is crucial in order to analyze sequential data. To capture these events automatically, we compute the optimal cluster count for each time step using the AICc criterion.

Using synthetic data, we demonstrate that our method exhibits a higher performance and is more robust to noise than previous work [4, 17]. We further show that our pipeline is able to extract stable clusters over time and reliably detects all cluster events. In an exploratory analysis on real-world data, we apply our pipeline to log data from two different ITS: One for spelling learning and one for mathematics learning. Finally, we present a set of visual tools that are powerful to analyze temporal data and student clusters.

2. METHOD

Our method for student clustering is designed to address two challenges when clustering temporal data. First, the method provides temporally consistent clusters. Second, our pipeline is able to capture changes in cluster sizes as well as in the number of clusters. Four *cluster events* are of particular interest in the context of educational data mining: merging, splitting, dissolving and forming of clusters. If the behavior of students from two different clusters becomes more similar over time, we expect the clusters to *merge* (this could mark a training effect). If on the other hand the behavior of students in a cluster sufficiently diverges clusters might *split* (this could mark the development of different learning strategies). If a distinct behavior disappears within a group of students, we assume the cluster will *dissolve*, meaning students will uniformly change to other clusters. In contrast, *forming* clusters have the potential to mark the development of distinct strategies within students.

The resulting clustering pipeline addressing these challenges is illustrated in Figure 1. The only input required are action sequences extracted from student log data. These action sequences are transformed into Markov Chains for every session and pairwise similarities between these chains are computed. Students are clustered based on these similarities while enforcing temporal consistency over consequent training sessions. Finally, we compute the optimal number of clusters for each training session.

Action Sequences. In a first step we extract action sequences $A_u^t = (a_0, a_1, \ldots, a_n)$ for every session t of a user u. To do so, we map events in the log files of an ITS (e.g. correct/incorrect inputs or help calls) to the actions a_i. As the particular actions depend on the ITS, the extraction of actions has to be changed depending on the ITS.

Action Processing. While action sequences provide rich temporal information about the exact ordering of actions, we expect that they exhibit a considerable amount of noise. We therefore transform the action sequences into an aggregated representation using Markov Chain models, similar to [17]. Markov Chains provide an aggregated view of the pairwise transition probabilities of actions and can be fully described by these transition probabilities $t_{i,j} := p_{a_j|a_i}$ from any state a_i (in our case an action) to any other state a_j. Markov Chains can be extracted using maximum likelihood estimates of the transition probabilities $t_{i,j}$.

Similarity Computation. To cluster student behavior, a suitable similarity (or distance) measure between students has to be defined. In educational data mining, popular choices for measuring distances between action sequences are the longest common subsequence (LCS) and the Levenshtein distance (see e.g. [4]). LCS measures the length of the largest set of characters that appear in left-to-right order within the string, not necessarily at consecutive places. The Levenshtein distance computes the number of insertions, deletions and replacements needed to transform one string into the other. Instead of computing distances directly on action sequences we can apply the computation to the aggregated values of Markov Chains. Previous work [17] has been using the Euclidean distance between the transition probabilities of two Markov Chains. A potential disadvantage of the Euclidean distance is that it is not designed for the comparison of probabilities. Therefore, we propose to use metrics that are specifically designed for comparing probability distributions. Since the conditional probabilities describing a Markov Chain do not form a proper probability distribution (the entries of the transition probability matrix do not sum up to one), we compute the expected transition probabilities using the stationary distribution over the actions and compare these expected transition frequencies $\bar{t}_{i,j}$ instead of the conditional probabilities $t_{i,j}$. We use two common metrics: the Jensen-Shannon Divergence and the Hellinger distance [21] to compute the distances between the expected transition frequencies $\bar{t}_{i,j}$ of the Markov Chains.

Clustering. Using the measures defined above we compute a pairwise similarity matrix W^t for every session t of the training (entries of the matrix measure how similar two students are during that particular training session). These similarity matrices can then be clustered by any standard clustering method. However, clustering students for each session individually does not make use of the temporal information available. Recently, a method for clustering evolutionary data has been proposed that accurately tracks the time-varying similarities of objects over discrete time steps [27]. The method assumes that the observed similarities W^t are a linear combination of the true similarity between students Ψ^t and random noise N^t:

$$W^t = \Psi^t + N^t. \tag{1}$$

Instead of performing clustering directly on W^t, a smoothed similarity matrix $\hat{\Psi}^t$ is proposed, given as

$$\hat{\Psi}^t = \alpha^t \hat{\Psi}^{t-1} + (1 - \alpha^t) W^t, \tag{2}$$

where α^t controls the amount of smoothing applied to the observed similarity matrix W^t. Under some assumptions (detailed in [27]) an optimal choice for α^t is

$$\alpha^t = \frac{\sum_i \sum_j \mathrm{var}(n_{ij}^t)}{\sum_i \sum_j (\hat{\psi}_{ij}^{t-1} - \psi_{ij}^t)^2 + \mathrm{var}(n_{ij}^t)}. \tag{3}$$

This means that the optimal α^t is based on a trade-off between the estimated noise in W^t and the amount of new information that W^t contains compared to previous similarity matrices. If W^t exhibits a lot of noise we more heavily rely on previous observations (high α^t) but if we observe large

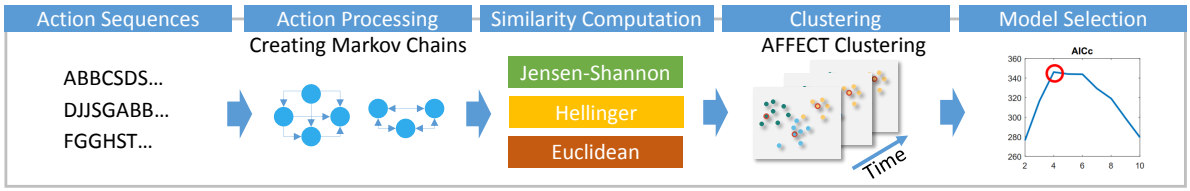

Figure 1: Overview of our clustering pipeline. Action sequences are extracted from log data and transformed into Markov Chains per session. Pairwise similarities between students are computed for every session. Clustering is performed using evolutionary clustering [27]. Finally, the AICc criterion selects the best model.

discrepancies between the previous similarity estimates and the current ones (e.g. some students show a novel behavior) we emphasize the similarities from the current session (low α^t). Finally, we use the standard clustering algorithm K-Means to cluster the smoothed similarity matrices $\hat{\Psi}^t$.

Model Selection. The assumption of temporal consistency in the pairwise similarities between students does not prohibit evolution of clusters if students change their behavior over the course of the training. Such long-term drifts lead to growing and shrinking of clusters eventually, and even to dissolving and forming of clusters over time. In contrast to the original AFFECT method [27], we therefore compute the optimal number of clusters in every time step. Deciding on the number of clusters is a variant of the model selection problem, for which various different criteria exists. The Akaike information criterion (AIC) and the Bayesian information criterion (BIC) are among the most common criteria for model selection. The main difference between BIC and AIC is that the BIC penalizes the number of clusters more strongly than AIC. AICc corrects the AIC criteria for finite sample sizes. For our experiments, we used AICc as it potentially reveals more clusters, which is important for our exploratory analysis of learning data. To compute the AICc the log likelihood (LL) of the model is needed. According to [23], the LL for K-Means can be formulated as

$$LL = \sum_i \log(\frac{N_{c(i)}}{N}\phi(x_i|\mu_{c(i)},\sigma)), \tag{4}$$

where N denotes the number of samples, $c(i)$ the cluster index of sample x_i and $N_{c(i)}$ the number of samples in cluster $c(i)$. The likelihood of a sample x_i that was assigned to cluster $c(i)$ can be computed using the probability distribution $\phi(x_i|\mu_{c(i)},\sigma)$, where $\mu_{c(i)}$ denotes the centroid of the cluster and σ the empirical variance of the data. In our case (as suggested by [23]), the probability distributions ϕ are identical spherical Gaussians. To compute the LL, we embed our data points in a Euclidean space in which the distances between the points match the similarities extracted from the action sequences. To perform this embedding, we use the method presented in [12] that transforms N objects with pairwise similarities to a $D = N - 1$ dimensional Euclidean space.We then estimate the effective dimensionality $\hat{D}$ of our data set as the sum of eigenvalues λ_i of the covariance matrix divided by the largest eigenvalue λ_1 (see [16]): $\hat{D} = \sum_i \lambda_i/\lambda_1$. This means that the effective number of parameters P for the K-Means clustering is $P = (\hat{D} + 1)k$, where k is equal to the number of clusters (see e.g. [23] for a derivation). Based on the LL and the estimated effective dimensionality of our data $\hat{D}$, we calculate the AICc as $-2LL + 2P + (2P(P+1))/(n-P-1)$.

3. SYNTHETIC EXPERIMENTS

We analyzed the properties of our clustering algorithm using synthetic data and we compared the performance and stability of our method to previous algorithms for clustering sequential educational data. Finally, we also validated our model selection step.

Experimental setup. We simulated student actions for 80 students over 50 sessions in a simulated learning environment. Students needed to solve 20 tasks per session. Student abilities θ and task difficulties d were simulated as part of a Rasch model [26]. Student abilities for all students were sampled from a normal distribution with mean μ and variance σ. Task difficulties were sampled uniformly from the range $[-3, 3]$ in agreement with the common range of task difficulties [10]. Each task y consisted of eight steps s_j that a student had to complete to finish the task (this could e.g. be letters of a word to spell, performing steps of a calculation or solving a physics problem). The probability of a student correctly solving a task was then given by the Rasch model as $p(y) = (1 + e^{-(\theta-d)})^{-1}$. In our simulation (in accordance with many ITS) a task was correctly solved if all the substeps are correctly solved, which defines the probability of correctly solving a step of a task s_j to be $p(s_j) = (p(y))^{\frac{1}{8}}$. Finally, a student could request help at any point in time during the training. Whether the student asked for help was sampled from a Bernoulli distribution with p_H. Based on the described sampling procedure we emitted the following actions for a student: *new task*, *help*, *correct*, *incorrect*, *correction*, *task completed*. The number of sampled actions per student and session depended on the performance of the student (e.g. a student who gets every step of a task correct completes a task after eight *correct* actions, whereas another student who requests help and commits an error requires more actions to complete the task).

For our experiments we simulated student groups with different behavior. For the chosen range of task difficulties, student abilities are found to be normally distributed with mean $\mu = 0$ and variance $\sigma = 1$ (see [10] for details). We simulated good performing students by setting $\theta = 1$ and bad performing students by setting $\theta = -1$. According to [2], the most frequent form of help abuse are multiple consecutive help requests. We simulated this behavior by a large probability $p_H = 0.2$ to ask for help instead of working on the task, while normal help seeking behavior has a smaller probability for requesting help $p_H = 0.05$. Based on these different properties we simulated four groups of 20 students as follows. Group A contains bad performing students ($\theta = -1$) that rarely ask for help ($p_H = 0.05$). Group B consists of bad performing students ($\theta = -1$) that frequently use the help system ($p_H = 0.2$). Group C and D consist of

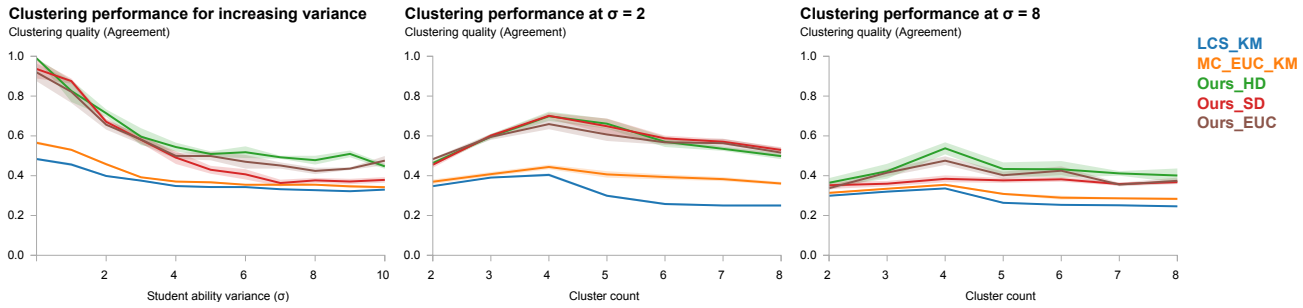

Figure 2: Comparison of clustering methods over increasing noise levels (left) and over different numbers of clusters for fixed noise levels $\sigma = 2$ (middle) and $\sigma = 8$ (right). Our method (*Ours_HD*, *Ours_SD*, *Ours_EUC*) shows less degradation of clustering quality (agreement with ground truth) for increasing noise levels.

good performing students ($\theta = 1$) with rare ($p_H = 0.05$) and frequent ($p_H = 0.2$) help requests, respectively.

Our proposed pipeline offers flexibility in the choice of the similarity measure (see Section 2). We used the Jensen Shannon divergence [21], the Hellinger distance [21] and the Euclidean distance for our experiments, and refer to these approaches as *Ours_SD*, *Ours_HD*, and *Ours_EUC*. To measure the influence of the different elements of the pipeline on the overall performance, we compared the proposed method to previous work on clustering of action sequences. The first approach [4] works directly on the action sequences and uses the longest common subsequences (LCS) as similarity measure. Clustering is performed using an agglomerative clustering. However, to be able to better compare clustering results we used the proposed similarity measure together with K-Means. We refer to this pipeline as *LCS_KM*. Similar to our method, the second approach used for comparison [17] computes the similarities between students using Markov Chains. Similarities are measured using the Euclidean distance and clustering is performed using K-Means. The pipeline for this approach is denoted by *MC_EUC_KM*.

Clustering Quality & Robustness. In a first experiment, we computed the clustering quality of the different approaches with increasing noise levels. The performance P was measured using the cluster agreement in comparison to the ground truth labels. The different noise levels were simulated by increasing the variance in student abilities σ for the sampling of the data. Figure 2 (left) illustrates the performance of the different approaches with increasing noise. Note that the performance was computed using the correct cluster count of $k = 4$. Our pipeline (colored in green, red, and brown) exhibits the highest performance over all noise levels. The average agreement of our best performing pipeline (P_{Ours_HD}) is substantially higher than the average agreement of the best previous approach ($P_{MC\_EUC\_KM}$), both for a low variance ($P_{Ours\_HD,\sigma=1} = 0.82$, $P_{MC\_EUC\_KM,\sigma=1} = 0.53$) and for noisy data ($P_{Ours\_HD,\sigma=10} = 0.45$, $P_{MC\_EUC\_KM,\sigma=10} = 0.34$).

To investigate these differences between the approaches, we measured their performance over different numbers of clusters at preset noise levels. Figure 2 (middle) illustrates the results for data with a relatively low noise level ($\sigma = 2$), while Figure 2 (right) shows the clustering quality of the different pipelines on noisy data ($\sigma = 8$). In the case of small noise in the data, all methods exhibit the best performance for the correct number of clusters ($k = 4$), which is a desirable property. The results demonstrate that using Markov Chains ($P_{MC\_EUC\_KM,k=4} = 0.44$) instead of working directly on action sequences ($P_{LCS\_KM,k=4} = 0.40$) leads to a higher clustering quality. A further increase in performance is achieved by our proposed algorithm: The variations of our pipeline exhibit a substantially higher clustering quality ($P_{Ours\_EUC,k=4} = 0.66$, $P_{Ours\_HD,k=4} = 0.70$, $P_{Ours\_SD,k=4} = 0.70$) than the previous work. This substantial increase in performance ($\Delta P_{k=4} = 0.26$ compared to MC_EUC_KM) is due to two changes in the pipeline. First, the proposed pipeline uses the AFFECT method for clustering leading to an increase in performance of $\Delta P_{k=4} = 0.20$. Second, while MC_EUC_KM computes the similarity measure directly on the transition probabilities, we use the expected transition probabilities as a basis for the similarity computations (see Section 2) accounting for an improvement in performance of $\Delta P_{k=4} = 0.06$. Within our approach, the choice of similarity measure has only a small impact on the clustering quality. Figure 2 (right) demonstrates that our proposed method is more robust to noise than previous work [17, 4]. The best variation of our pipeline (colored in green) still achieves a reasonable performance ($P_{Ours\_HD,\sigma=8} = 0.54$). At these noise levels, the choice of action processing (Markov Chains vs. direct processing of action sequences) does not significantly influence performance ($P_{LCS\_KM,k=4} = 0.34$, $P_{MC\_EUC\_KM,k=4} = 0.35$). The choice of the clustering algorithm on the other hand is important. The increased performance of our method can be attributed to the use of AFFECT for clustering: AFFECT takes into account data from previous time steps to perform the clustering. Interestingly, the pipeline using the Jensen Shannon divergence (*Ours_SD*) seems less robust to noise than the other pipelines (*Ours_HD* and *Ours_EUC*).

Stability. When clustering student actions over time, temporal consistency of clusters is essential. We measured the temporal stability of our method by computing the cluster size over the 50 simulated sessions (see Figure 3). We compared the best performing pipeline from the first experiment (*Ours_HD*) to the previous approaches (*LCS_KM*, *MC_EUC_KM*) using again $k = 4$ clusters. As can be seen from Figure 3 (left), our method provides a smooth temporal clustering with stable cluster sizes over time. The clusters found by *MC_EUC_KM* (Figure 3 (middle)) and *LCS_KM* (Figure 3 (right)), on the other hand, are unstable: cluster sizes vary significantly over time. These results are as expected, as static clustering approaches identifying groups of students at each point in time are very sensitive to

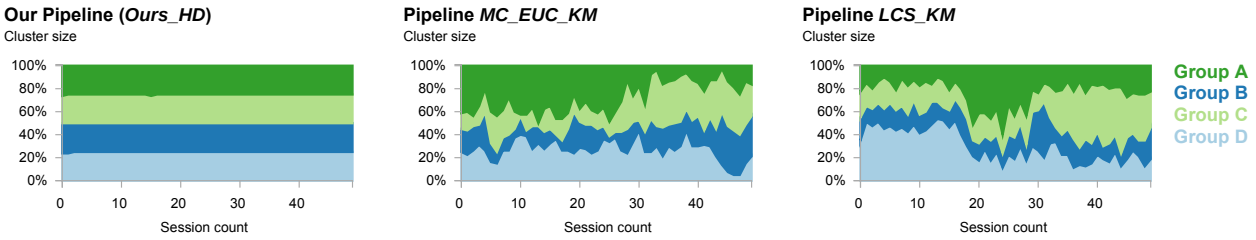

Figure 3: Relative cluster sizes (for $k = 4$ clusters) over 50 simulated sessions. Our method performs best in extracting temporally stable clusters.

noise. The proposed method solves this problem by applying an evolutionary clustering algorithm and therefore takes into account multiple time steps.

Interpretability. Since we are clustering student behavior over multiple sessions, we expect the number of clusters and the cluster sizes to change over time. We expect clusters to merge, split, dissolve and form (see Section 2 for details). We evaluated the *Ours_HD* pipeline on four scenarios using synthetic data. Note that these scenarios are artificial and are used only to demonstrate that the pipeline can capture the described events; we will show real-world examples of these events in Section 4. In the first scenario (Figure 4 (top left)), group A consisting of bad performing students with rare help calls (colored in dark green) merges into group B (colored in dark blue), i.e. the students of group A also start abusing the help. In our simulation, we start the cluster merge after $t = 20$ sessions and let group A completely vanish after $t = 50$ sessions, a behavior that is nicely captured by our method. The second scenario (Figure 4 (top right)) starts with only three groups (B, C, and D), assuming that all bad performing students frequently use the help. Over time, the bad performing students split into a group abusing the help (group B, colored in dark blue) and a cluster consisting of students with rare help calls (group A, colored in dark green), i.e. in the simulation some of the bad performing students stop abusing the help over time. In the third scenario (Figure 4 (bottom left)) a dissolving cluster is simulated: Over time, group B (colored in dark blue) completely dissolves and the students are distributed to the other three clusters. The fourth scenario (Figure 4 (bottom right)), finally, simulates a forming cluster event. The simulation starts with only three clusters (groups A, C, and D). With an increasing number of sessions, a fourth cluster forms (group B, colored in dark blue) and students from the other three clusters slowly switch to the new cluster until all the groups have equal size (after $t = 50$ sessions). This event is again correctly captured by our method. The presented experiments demonstrate that the proposed pipeline is able to reliably identify changing cluster numbers and sizes. The results also demonstrate the validity of the model selection step of the pipeline: The AICc correctly identifies the number of clusters for all scenarios.

4. EXPLORATORY DATA ANALYSIS

We applied our method to clustering of student interactions from two different ITS, focusing on the identification and interpretation of *cluster events*.

Experimental Setup. The first data set contains log data from 106 students and was collected using *Orthograph*, a

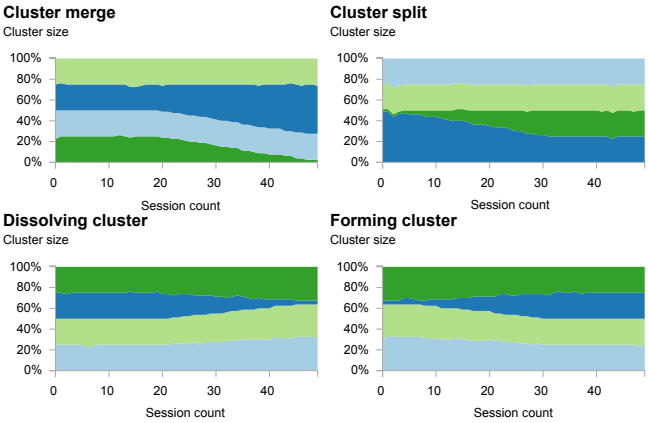

Figure 4: Simulated examples of four types of *cluster events*. Our pipeline correctly identifies cluster merges/splits as well as dissolving/forming clusters.

computer-based training program for elementary school children with dyslexia [9]. *Orthograph* consists of one main learning game, where children have to type a dictated word. The second data set contains data from 134 students and was collected from *Calcularis*, an ITS for elementary school children with difficulties in learning mathematics [13]. *Calcularis* consists of different games for training number representations and calculation. For all students, we extracted the first 15 training sessions with a minimal duration of $t = 5$ minutes from each student.
All results have been computed using our pipeline *Ours_HD* (see Section 2), applying the Hellinger Distance to measure similarities between Markov Chains of different students.

Navigation Behavior. In a first experiment, we extracted actions describing the *Navigation Behavior* of children in *Orthograph*. *Navigation Behavior* captures all events that cause the displayed content to change. During game play, children collect points for correct responses as well as for time spent in the training in general. These points can be used to buy different visual perks for the game in the shop. Children can also analyze their performance (e.g. progress in the current module) in the progress view. The resulting Markov chain (see Figure 5) consists of three possible states: *Game*, *Shop*, and *Performance*.

Figure 6 shows the relative cluster sizes for the *Navigation Behavior* Markov Chain over the first 15 sessions of the training. The different colors denote different clusters. At the beginning of the training ($t = 0$), our pipeline detects seven different clusters, however, three of these clusters (col-

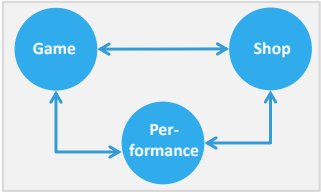

Figure 5: Markov Chain for actions that capture the *Navigation Behavior* of students in *Orthograph*.

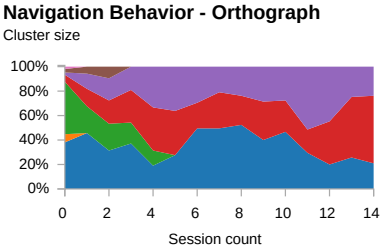

Figure 6: Relative cluster sizes based on the *Navigation Behavior* extracted from *Orthograph*.

ored in pink, brown, and orange) die within the first three training sessions. Children in these clusters spent more than 50% of their time browsing the shop and checking their performance (orange: 46% *Game*, 31% *Shop*, 23% *Performance*; brown: 43% *Game*, 22% *Shop*, 35% *Performance*; pink: 40% *Game*, 32% *Shop*, 28% *Performance*) at the beginning of the training. We therefore hypothesize that children in these clusters tried out and played with the different views before getting used to the navigation possibilities of the system.

After $t = 5$ time steps, a further cluster (colored in green) dissolves before the clustering stabilizes to three main groups (colored in blue, red, and purple). Figure 7 (top) shows the transition probabilities of the Markov Chains for the different clusters before the clusters dissolve (after $t = 3$ sessions). Children in the blue cluster are very focused on training, they spend 82% of their time in the *Game*. Once in the *Shop* or *Performance* state (18% of their time) they tend to select the following view with equal probabilities. Children in the red cluster like to browse the shop, a behavior that is visible from the high transition probabilities to the *Shop* state (*Game*→*Shop*: 0.41; *Performance*→*Shop*: 0.39), resulting in 34% of the training time spent browsing the shop. The purple cluster consists of children, who like to navigate to the shop and performance overview between solving the different tasks (*Game*→*Shop*: 0.41, *Game*→*Performance*: 0.44). However, these tend to be shorts visit as they will return to playing the game right after with high probability (*Performance*→*Game*: 0.58, *Shop*→*Game*: 0.77). Finally, children in the green cluster tend to select the next view randomly when playing the game. Once in the *Performance* state, they have a probability of 0.30 to browse the shop right after. The analysis of this time step illustrates that the different clusters differentiate well between focused children not making use of the navigation possibilities (blue cluster), children who frequently (but reasonably) use the different views (purple and green cluster), and distracted children who spend long amounts of time off-task (red cluster).

After $t = 6$ training sessions, the green cluster dissolves and students from this cluster change to the red and blue clusters. The transition probabilities of the Markov Chains for these stable main clusters are illustrated in Figure 7 (bottom). The children in the blue cluster are still focused on training, spending 76% of their time solving tasks. However, they also check their training progress from time to time (14% of the time spent in the *Performance* state). After checking training progress, they tend to also browse the shop (*Performance*→*Shop*: 0.27). The children in the purple cluster have stopped navigating to the performance overview between different tasks (*Game*→*Performance*: 0.17) and instead visit the shop more frequently (*Game*→*Performance*: 0.58) and longer (35% of time spent in the *Shop* state). The red cluster still consists of children who like browsing the shop, a behavior that is visible from the high transition probabilities to the *Shop* state (*Game*→*Shop*: 0.33; *Performance*→*Shop*: 0.31). However, they also tend to spend time checking their progress, resulting in 47% of the training time spent off-task. Students from the green cluster therefore changed their behavior from frequent, but short off-task navigation to a more focused training style (change to blue cluster) or to being completely distracted and spend long amount of times off-task (change to the red cluster).

Input & Help Seeking Behavior. Our method can be used as a black box for any ITS and therefore also allows for comparison of behavior patterns across different ITS. The only user input needed is the definition of possible actions. To illustrate this possibility, we extracted two different sets of actions *Input Behavior* and *Help Seeking Behavior* from data collected with *Orthograph* and with *Calcularis*.

Input Behavior captures all possible inputs. Implicitly these actions capture the performance of students, as e.g. a bad performing student is likely to commit more mistakes. In *Orthograph*, children train spelling by writing words that are played back by the system. Therefore, the *Input Behavior* Markov Chain for *Orthograph* (see Figure 8) consists of four states: Children can type a letter (*Input*), correct themselves by deleting a letter (*Backspace*), provide invalid input such as typing a number (*Invalid Input*), or submit their solution (*Enter*). For *Calcularis*, we investigated calculation games. In these games, children need to solve different mental addition and subtraction tasks. We again define four states for the *Input Behavior* Markov Chain (see Figure 8): children can type a digit (*Input*), correct themselves by deleting a digit (*Correction*), provide invalid input such as random mouse clicks (*Invalid Input*), or set their answer (*Enter*).

Figure 9 shows the relative cluster sizes for the *Input Behavior* action set from *Orthograph* over 15 training sessions. Our method identifies three stable clusters. Investigating the stationary distributions of the Markov Chains reveals that students in the orange cluster show the highest probabilities for committing invalid inputs over all sessions ($t=3$: 0.15; $t=7$: 0.23; $t=13$: 0.16). The green cluster consists of focused students who consistently produce a low percentage of invalid inputs ($t=3$: 0.06; $t=7$: 0.04; $t=13$: 0.05). Students in the blue cluster also tend to show low probabilities for invalid inputs across the different sessions ($t=3$: 0.11; $t=7$: 0.09; $t=13$: 0.08). The orange cluster is an example of a *forming cluster* growing in size over the course of the training. We hypothesize that this event marks the increasing difficulty of the tasks and is caused by a downwards drift

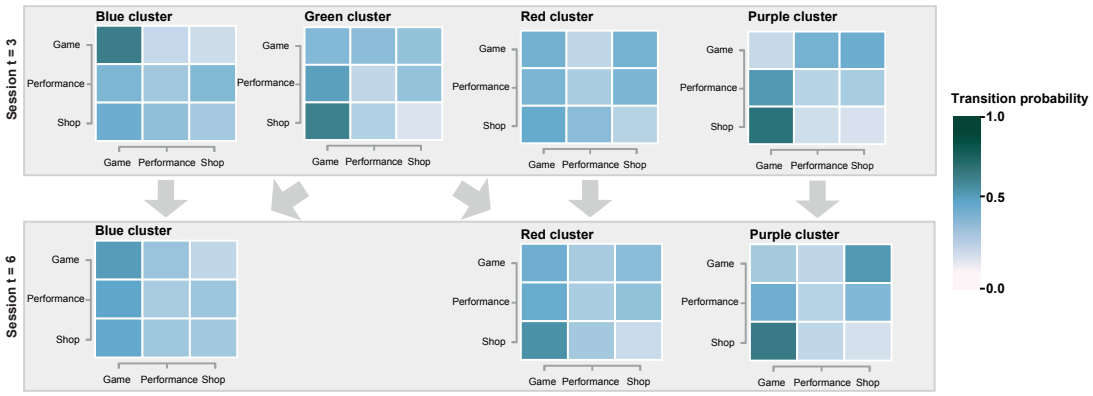

Figure 7: Transition probabilities of the average Markov Chain for each cluster present in session $t = 3$ (top) and session $t = 6$ (bottom) for *Navigation Behavior* in *Orthograph*. The arrows indicate students transferring from the green cluster to the blue and red clusters between session $t = 3$ and session $t = 6$.

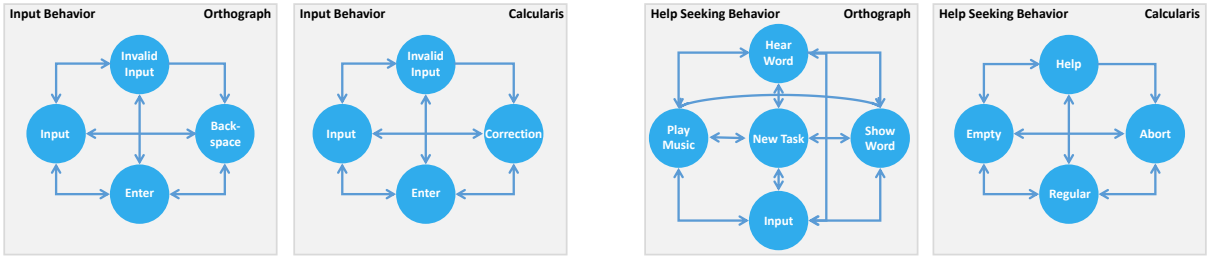

Figure 8: Markov Chains for the *Input Behavior* and the *Help Seeking Behavior* in *Orthograph* and *Calcularis*.

of students from the clusters with good performing students to the clusters with students showing worse performance. Further analysis of cluster transfers reveals that students indeed are never switching directly from the green (best performance) to the orange cluster (worst performance).

For *Calcularis*, the *Input Behavior* clusters are relatively stable over the course of the training (see Figure 9). There is one distinct *dissolve event* in the first four sessions: the orange cluster is dissolving into the blue and green clusters. Investigating the stationary distributions of the Markov chains of the three clusters reveals that all clusters have a relatively low probability for invalid inputs (t=2: 0.17 (blue), 0.12 (orange), 0.08 (green)). However, students belonging to the blue cluster tend to perform multiple consecutive corrective actions in a row (*Correction*→*Correction*: 0.25 (blue), 0.13 (orange), 0.13 (green)). Students in the orange cluster are most likely to enter a valid input after a correction (*Correction*→*Input*: 0.68 (orange), 0.57 (blue), 0.65 (green)).

In *Orthograph*, differences in *Input Behavior* are mainly expressed by the percentage of invalid inputs provided. We observe a more distinct picture for *Calcularis*. While the invalid inputs are still an important indicator, children also exhibit different corrective behaviors.

Help Seeking Behavior captures the use of hints available in the training environment. In *Orthograph*, children can re-play the given word (*Hear Word*), play the melody of the word (*Play Music*) and show the correct spelling of the word (*Show Word*). The according Markov Chain is displayed in Figure 8. The states *New Task* and *Input* denote the play-back of a new word and a user input (keyboard), respectively. The development of the relative cluster sizes for these action sequences (see Figure 9) reveals a surprisingly large variance in student behavior (the clustering algorithm finds nine different clusters in the first two training sessions). However, the diversity in student behavior disappears through a large *cluster merge* after $t=3$ sessions. Investigating the transition probabilities between the different actions, we observe that while students are experimenting with the three different help systems at the beginning of the training, the final cluster of students gave up on using the help functions. This drop in the frequency and diversity of help usage indicates that the help functionality provided in *Orthograph* is not useful for most of the students.

Calcularis provides a limited help functionality. Children can require explanations for games (*Help*). Furthermore, they can directly require the solution of a task (*Empty*), if the task seems too difficult. Further states of the Markov Chain (displayed in Figure 8) are the setting of a complete answer (*Regular*) and the abortion of a task (*Incomplete*). We again observe a large *cluster merge* at the beginning of the training leading into two stable clusters. Investigating the stationary distributions of the Markov Chains of the two clusters reveals that students in the orange cluster are more likely to perform a help request compared to the blue cluster (t=6: 0.03 (blue), 0.13 (orange)).

The *Help Seeking Behavior* of the children is more difficult to compare across different ITS, because the available hints are very different. However, our experiment shows that both learning environments do not provide ideal help options.

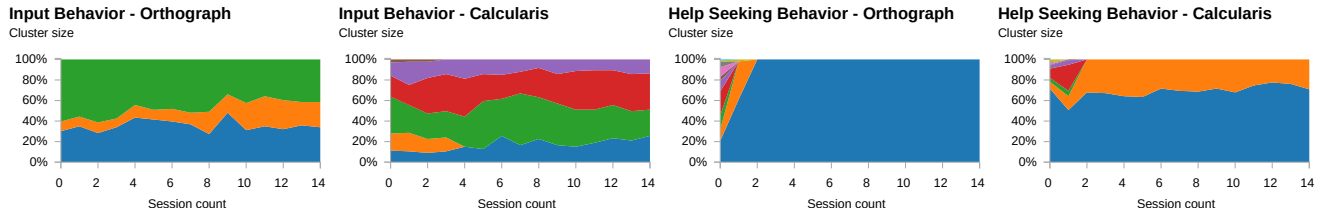

Figure 9: Relative cluster sizes over the first 15 sessions based on the clustering of *Input Behavior* (left) and *Help Seeking Behavior* (right) for students training with *Orthograph* and *Calcularis*.

5. CONCLUSIONS

We presented a complete pipeline for the evolutionary clustering of student behavior. This pipeline can be used as a black box for any ITS, requiring only the extraction of action sequences as input. We demonstrated that enforcing temporal coherency between consecutive clusterings is beneficial for the detection of student behavior as well as the stable detection of *cluster events*. Our method outperforms previous work on synthetic data regarding clustering quality and stability. We applied our pipeline to different types of action sequences collected from two different ITS. The exploratory analysis demonstrates that our method is able to reveal interesting properties about the behavior of students and potential deficiencies of the learning environments.

Acknowledgments. This work was supported by ETH Research Grant ETH-23 13-2.

6. REFERENCES

[1] R. Agrawal and R. Srikant. Mining sequential patterns. In *Data Engineering*. IEEE, 1995.

[2] V. Aleven, B. McLaren, I. Roll, and K. Koedinger. Toward meta-cognitive tutoring: A model of help seeking with a cognitive tutor. *IJAIED*, 2006.

[3] J. M. L. Andres, M. M. T. Rodrigo, R. S. Baker, L. Paquette, V. J. Shute, and M. Ventura. Analyzing Student Action Sequences and Affect While Playing Physics Playground. In *AMADL*, 2015.

[4] Y. Bergner, Z. Shu, and A. A. Von Davier. Visualization and Confirmatory Clustering of Sequence Data from a Simulation-Based Assessment Task. In *Proc. EDM*, 2014.

[5] G. Biswas, H. Jeong, J. S. Kinnebrew, B. Sulcer, and R. ROSCOE. Measuring self-regulated learning skills through social interactions in a teachable agent environment. *RPTEL*, 2010.

[6] K. E. Boyer, R. Phillips, A. Ingram, E. Y. Ha, M. Wallis, M. Vouk, and J. Lester. Characterizing the Effectiveness of Tutorial Dialogue with Hidden Markov Models. In *Proc. ITS*, 2000.

[7] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. In *Proc. KDD*, 2006.

[8] M. Desmarais and F. Lemieux. Clustering and visualizing study state sequences. In *Proc. EDM*, 2013.

[9] M. Gross and C. Vögeli. A multimedia framework for effective language training. *Comput. & Graph.*, 2007.

[10] D. Harris. Comparison of 1-, 2-, and 3-parameter IRT models. *Educational Measurement*, 1989.

[11] J. Herold, A. Zundel, and T. F. Stahovich. Mining meaningful patterns from students' handwritten coursework. In *Proc. EDM*, 2013.

[12] T. Hofmann and J. M. Buhmann. Pairwise data clustering by deterministic annealing. *Pattern Analysis and Machine Intelligence*, 1997.

[13] T. Käser, G.-M. Baschera, J. Kohn, K. Kucian, V. Richtmann, U. Grond, M. Gross, and M. von Aster. Design and evaluation of the computer-based training program calcularis for enhancing numerical cognition. *Frontiers in Developmental Psychology*, 4(489), 2013.

[14] J. S. Kinnebrew and G. Biswas. Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution. In *Proc. EDM*, 2012.

[15] J. S. Kinnebrew, D. L. Mack, and G. Biswas. Mining temporally-interesting learning behavior patterns. In *Proc. EDM*, 2013.

[16] M. Kirkpatrick. Patterns of quantitative genetic variation in multiple dimensions. *Genetica*, 2006.

[17] M. Köck and A. Paramythis. Activity sequence modelling and dynamic clustering for personalized e-learning. *UMUAI*, 2011.

[18] C. Li and G. Biswas. A Bayesian Approach to Temporal Data Clustering using Hidden Markov Models. In *ICML*, 2000.

[19] R. Martinez-Maldonado, K. Yacef, and J. Kay. Data mining in the classroom: Discovering groups' strategies at a multi-tabletop environment. In *Proc. EDM*, 2013.

[20] L. D. Miller and L.-K. Soh. Meta-Reasoning Algorithm for Improving Analysis of Student Interactions with Learning Objects using Supervised Learning. In *Proc. EDM*, 2013.

[21] L. Pardo. *Statistical inference based on divergence measures*. CRC Press, 2005.

[22] T. Peckham and G. McCalla. Mining Student Behavior Patterns in Reading Comprehension Tasks. In *Proc. EDM*, 2012.

[23] D. Pelleg and A. Moore. X-means: Extending K-means with Efficient Estimation of the Number of Clusters. In *Proc. ICML*, 2000.

[24] D. Perera, J. Kay, I. Koprinska, K. Yacef, and O. R. Zaïane. Clustering and sequential pattern mining of online collaborative learning data. *TKDE*, 2009.

[25] Y. Wang and N. T. Heffernan. The student skill model. In *Proc. ITS*, 2012.

[26] M. Wilson and P. De Boeck. Descriptive and explanatory item response models. In *Explanatory item response models: A generalized linear and nonlinear approach*. Springer, 2004.

[27] K. S. Xu, M. Kliger, and A. O. Hero Iii. Adaptive evolutionary clustering. *Data Mining and Knowledge Discovery*, 2014.